Copyright

by

Dustin Lee Smith

2014

The Thesis Committee for Dustin Lee Smith
Certifies that this is the approved version of the following thesis:

Steganoscription: Exploring Techniques for Privacy-preserving Crowdsourced Transcription of Handwritten Documents

APPROVED BY
SUPERVISING COMMITTEE:

Supervisor:

_____
Unmil Karadkar

_____
Pat Galloway

Steganoscription: Exploring Techniques for Privacy-preserving Crowdsourced Transcription of Handwritten Documents

by

Dustin Lee Smith, B.S.

Thesis

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Information Studies

The University of Texas at Austin

May 2014

Dedication

I dedicate my master thesis to my wonderful wife, Christine. You've been an unwavering pillar of support throughout the nine month process that culminated in this master thesis. You supported me through countless late nights, early mornings, boring weekends, and stressful holidays. You were gracious when I would transition directly from my daily work to thesis work and you didn't get to see me at all or spend time with me during the 80-90 hour work weeks those last few months. I could not have made it without your love and support. I will never forget it. I love and cherish you!

Acknowledgements

I would like to express my gratitude to my supervisor Unmil Karadkar for introducing me to the topic, for his continual guidance, for his persistence in challenging my perceptions, and for his thorough comments and feedback on this master thesis. Furthermore I would like to thank my second reader, Pat Galloway, for her exciting character during our meetings, unique and artistic perspective on a very technical topic, and her comments and feedback on this master thesis. Also, I would like to thank Dr. King Davis, Celeste Henery, and Lorrie Dong for their support of this research and their advocating efforts to market it. Lastly I would like to acknowledge the National Association of State Mental Health Program Directors, the Substance Abuse and Mental Health Services Administrator, and the Institute for Urban Policy Research and Analysis, UT Austin as the primary supporting groups for this master thesis. I would like to thank my team at BuildASign.com, friends, and family, who have supported me throughout the entire process, both by keeping me sane and encouraging me along the way.

Abstract

Steganoscription: Exploring Techniques for Privacy-preserving Crowdsourced Transcription of Handwritten Documents

Dustin Lee Smith, M.S.Info.Stds.

The University of Texas at Austin, 2014

Supervisor: Unmil Karadkar

The focus my research is the historical document format represented by the Central State Hospital (CSH) dataset, handwritten medical records. The specific problem innate to the CSH dataset in question is how to transcribe sensitive, cursive-handwritten documents via a manual vehicle- such as crowdsourcing. Manual methods are necessarily no matter the sophistication of the optical character recognition system used because of the inconsistencies within cursive script. To address this problem I've developed an application that enables users to transcribe sensitive, handwritten, document images while preserving the privacy of the context around the transcribed text via random word selection and visual manipulation of the displayed text. This is made possible through several algorithms that process documents from a top-down approach. These system operations detect and segment lines of text in images, reverse the slant common to cursive script, detect and segment words, and finally, manipulate word-images before they are displayed to users; combinations of color, noise, and geometric manipulations

are currently supported and used randomly. This system, called Steganoscription, combines the concepts of steganography and transcription.

Table of Contents

List of Figures

Chapter 1 Introduction

BACKGROUND

As a society we learn from historical experiences to inform our future decisions—a reality which makes historical records a priceless artifact as well as a reminder of events. While much contemporary information can be found via Google searches, our historical records are unfortunately inaccessible at scale. This is because much of history is locked in physical documents that have not been indexed for search and analysis. As a society we have continuously improved on our ability to preserve these documents through robust archival practices and, more recently, digitization; the next step must be taken to convert these documents into a machine-readable format so that they can be indexed for search and analyzed at scale. For cursive documents, techniques that automatically detect words and transcribe them are not 100% viable, which means that crowdsourced transcription must be seriously considered. In the case of mental health documents, however, public transcription is not an option as the act of transcription involves reading them, which has implications for privacy of patients, doctors, and employees alike, potentially resulting in social repercussions to family and descendants. My research aims to apply mechanisms to crowdsource the transcription of privacy-sensitive documents while maintaining the anonymity of individuals named in handwritten records by constraining the context of such mention as well as by exploiting document characteristics. In so doing I aim to bridge the gap between the handwritten, textual documents that are inaccessible for machine processing and the high powered analysis tools that are readily available today. This thesis reports on the design and evaluation of a system for crowdsourced transcription of privacy-sensitive, cursive, handwritten, historical documents.

CENTRAL STATE HOSPITAL (CSH)

Central State Hospital was founded in 1870 as the first African-American mental hospital; the hospital is still active and is located in Petersburg, Virginia.[1] It was originally named Central Lunatic Asylum for the Colored Insane. Administrators of the CSH have kept and maintained meticulous records ever since its inception. These documents have recently been digitized into image format, but neither the physical nor the digitized documents are stored in an archival environment. This terabyte scale image dataset provides a great opportunity to explore transcription methods that will address issues likely to be encountered by many other historical collections of mental health records as similar hospitals were opened in other states soon after the Civil War.

CSH COLLECTION

The dataset of images includes handwritten cursive documents to record board meeting minutes, patient admission, treatment, and discharge, sign-in sheets, and reports from the hospital's director to the state governor. The research reported in this thesis has used the board meeting minutes as these minutes are considered institutional records rather than patient records and have the lowest possibility of identifying individual patients. As such, these minutes present the safest approach to avoid sensitive content and provide the variability that we need to cover a significant portion of the document structures. The documents are mostly handwritten, but also include printed format as well. This research only focuses on digitized handwritten documents.

There are about 500,000 professionally digitized pages across the entire dataset. The digital masters of these pages are 400 dpi TIFFs that cumulatively occupy 14

[1] http://www.csh.dmhmrsas.virginia.gov/about.html

terabytes of hard drive space and are stored in a folder structure that reflects minimal structural information.

OBJECTIVE

The objective of my research is to provide access to these records, especially text within the record for searching and finding topics, finding patterns in the data for a variety of demographics.

Chapter 2 Related Work

This research builds upon advances made in the following areas: off-line and online script recognition, document analysis, and Steganoscription and is presented in the context of these fields.

Off-line cursive recognition differs from on-line in that on-line recognition is performed at the time of writing and typically makes use of special writing surfaces; these surfaces track the writer's strokes and record points for later analysis. Off-line recognition is performed on pre-written text via many different approaches. Vinciarelli (2002) provides an extensive survey on cursive word recognition, both on-line and off-line, while Plamondon and Srihari (2000) present a broader view of handwriting recognition systems that are not specific to cursive text. My research method falls under the off-line classification of script recognition and like other approaches in this class; it includes document analysis, preprocessing, and segmentation.

DOCUMENT ANALYSIS

The field of document analysis focuses not on the contents of a document image, but on its structure and content layout. For many approaches in word segmentation, this is the first step and is followed by line detection and line segmentation, which involves locating lines of text within a document, separating individual lines from the document for further analysis and manipulation. Breuel (2003) introduces several algorithms for performing document layout analysis. Most notably and related to my research is his approach for locating lines of text. His method uses background analysis and white-space analysis to locate columns of text and perform local methods of line detection within the column to mitigate error introduced by multi-column documents. While the result of his

approach is not related to my work—in that my work does not involve columns, the method he uses to locate these columns is very similar. Background and white-space analysis are integrated steps of the histogram approach described in this thesis.

CURSIVE SCRIPT ANALYSIS

Cursive script was a focal point starting with the earliest research due to its dominance as a writing style for hundreds of years and even until the most recent decade. Mermelstein and Eyden (1964) have published the earliest known work in automatic on-line recognition of cursive script. The earliest off-line work was submitted about ten years later by Sayre (1973), which used letter n-grams and introduced the notion of letter segmentation. More recently Bozinovic and Srihari (1989) introduced several new techniques to be used for off-line cursive recognition, including methods for improving detection of letters and increasing efficiency of processing by reducing the number of passes through the document. Guillevic and Suen (1995) introduce methods for performing off-line cursive recognition on French bank checks. They present methods for dealing with limited vocabulary and free form cursive (not constrained to a line). Tomai et al. (2002) introduce a method for mapping cursive word-images to pre-transcribed words. Unfortunately the transcription of large datasets is an often-encountered barrier to post-transcription actions. Though, when these requirements are met, document analyst could benefit greatly from linking the text back to the original document. My software utilized the Gamera library for custom recognition systems; Droettboom et al. (2003) introduce this library, framework, and methods for character recognition that use the

software. De Zeeuw (2006) performs line and word segmentation as well as word slant correction using a histogram based method. Fischer et al. (2010) focus on historical documents and use a Hidden Markov Model (HMM) based word segmentation method that was introduced by Zimmermann and Bunke (2002). The strength of Zimmermann and Bunke's method is that it does not require the letter recognition step, which reduces the computational cost on the process.

Among the most crucial, and certainly the most unique, of preprocessing steps for cursive document segmentation is slant and/or skew correction. "Slant" refers to a word's deviation from the horizontal word baseline, shown in Figure 1b. Figure 1a illustrates a "skew" (or "shear") effect, which is the deviation of the vertical median of each letter with respect to the letter's base position. When persistent across the entire word, the effect creates the forward or backward angle common to the cursive script.

Figure 1: A comparison of forward sheared text (a) on the left and upward slanted text (b) on the right

The characteristic shear of cursive handwriting presents a unique challenge for automated recognition methods because of the variability of shear angle that is heavily influenced by the writer's style. In addition to the slant and shear correction methods introduced and

used by Vinciarelli (2002, Plamondon et al. (2000), Droettboom et al. (2003) and de Zeeuw (2006), Dong et al. (2005) perform Radon transformations to de-skew and de-slant word images based on stroke length and deviation from a word's baseline position.

STEGANOGRAPHY

To incorporate security into my research I apply steganography. Morkel et al (2005) define steganography as "the art of hiding the fact that communication is taking place, by hiding information in other information." My application of steganography stretches this definition. I am not hiding communication to an end user, but instead I am hiding the context of a message and its meaning within a larger context in order to communicate only the carrier of a message and not the message itself. Morkel et al. (2005) provide an overview of traditional image steganography definitions and techniques. A very popular application of image manipulation techniques, though not necessarily for steganography, is performed by Von Ahn et al. (2008), who introduced the reCAPTCHA system. This system is used on "over 200,000" websites to protect them from malicious automated scripts and robots by providing a security barrier that requires the translation of a distorted word-image into text.[2] Another key function is to use "wasted human

[2] https://www.google.com/recaptcha/admin#whyrecaptcha

processing power" to translate words within books that automated methods cannot recognize. Their system alters the words that are translated through techniques such as distorting the image, changing the aspect ratio, or altering the image with noise. These techniques serve to keep robots from reading the text, but can also be used to anonymize words from their contextual location such that neither humans nor robots can link the words together and discover the information that is within the full document. This is accomplished by masking the location of words within the larger document to which they belong. Without knowing the surrounding words or being able to match the background of a word to another word, it's unlikely that any meaning can be derived from the word. Removing the background context prevents users from associating words through document characteristics and distorting words prevents users from associating words based on script characteristics.

Chapter 3 Approach

In order to make the test in the document available for machine analysis, my approach uses a crowdsourcing technique rather than developing robust algorithms for text detection. Automated transcription for any text has not been perfect, and a human reader is often necessary to decipher at least some of the text. I am also designing for sensitive materials, such as health records, so a necessary function of my application is randomized word-image selection and image manipulation prior to displaying the image for the human transcriber. Figure 2 depicts the process from a high level; each of these steps will be discussed in the following sections of this chapter.

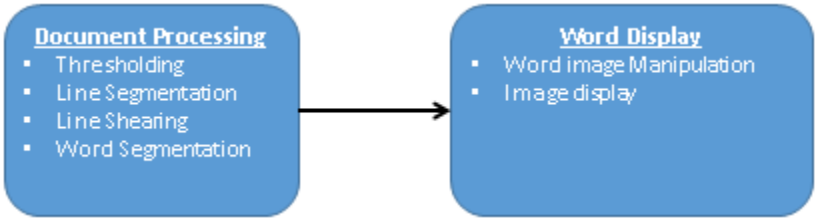

Figure 2: Order of algorithm processes.

GAMERA

The Gamera framework is a system, including a library and application, for document analysis. It also provides a basis on which users can build optical character recognition (OCR) systems upon. In research to date, this framework has primarily been used to detect and identify musical notations on handwritten bars. I vetted similar systems, such as Tesseract-OCR, but as past applications of Gamera have focused towards free-form scripts like musical notation and ancient glyphs it seemed like a better fit. Further

evaluation of Tesseract vs. Gamera will be discussed in the Discussion chapter. My initial approach was to use open source OCR and image libraries to segment the document images into words and for shearing these words. I expected the Gamera library[3] to carry the heaviest burden. In practice, Gamera's document analysis features including its projection and thresholding functions proved inadequate due to their dependency on the native datatype, ONEBIT; other libraries like scikit-image (skimage) and numpy are not compatible with this format. My revised process uses the projection functions and Gamera's implementation of DjVu thresholding, which are described in the following sections. After discussing thresholding and line segmentation I discuss the rest of my approach: image shearing, word segmentation, and word transcription.

THRESHOLDING

Thresholding is the first operation applied to a document image before my script can find line breaks. Since images are taken in full color, my script cannot accurate distinguish lines at the pixel level. To remove the spectrum of colors in the image, thresholding is applied. Simply put, thresholding takes variable inputs to force an image into binary: meaning every pixel either has a value of zero or one. There are several ways to execute thresholding even within the Gamera libraries; I chose DjVu after comparing the methods because it produced the clearest black and white image. Here I define "clearest" to mean that the words are fully black and not speckled with white pixels; also, the white around the words and throughout the image is consistent with the least amount of noise. DjVu was introduced by Haffner et al. (1999) for compression of high-quality images for slow-

[3] http://gamera.sourceforge.net/)

connection transfers. Part of their research led to a method for this bi-level image compression, converting an image into black/white pixels only. An example of the conversion of a full color image, as shown in Figure 3a, to a binary image, as shown in Figure 3b, using DjVu thresholding is given.

Figure 3: A comparison of colored text (a) on the left to binary threshold text (b) on the right

To do this while keeping the contents of the image intact and legible, their work uses a k-means algorithm to group the black pixels and white pixels appropriately such that the foreground and background of the image remain distinct.

LINE SEGMENTATION

Gamera's projection functions, projection_rows() and projection_cols() provide crucial analysis for detecting line breaks in document images. These functions count black pixels in every row/column within the document. Figure 4 shows an example of the function projection_rows being applied to a document. Longer lines contain more black pixels and thus produce a larger spike on the graph. For line detection I apply this projection_rows function to an entire document to obtain a vector of pixel densities at each row.

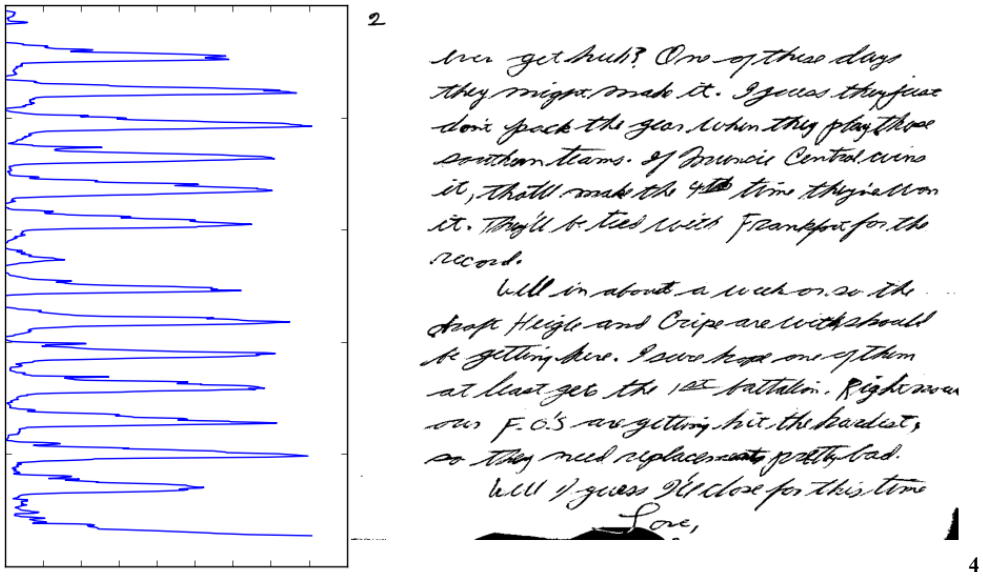

Figure 4: Visualization of applying the row projection function to a thresholded document.

With the row-density vector the algorithm I developed locates local minima within the document by using sliding windows, which translates the problem from finding local minima on a large document to finding absolute minima on a restricted view of the document. This transformation has the potential of producing incomplete or false lines, so my script mitigates this problem by requiring a minimum distance between minima. An example of the results produced by this approach is in Figure 6 compared to the original document in Figure 5.

[4] http://koreanwarletters.blogspot.com/2011/04/korean-war-letter-1st-marine-division.html

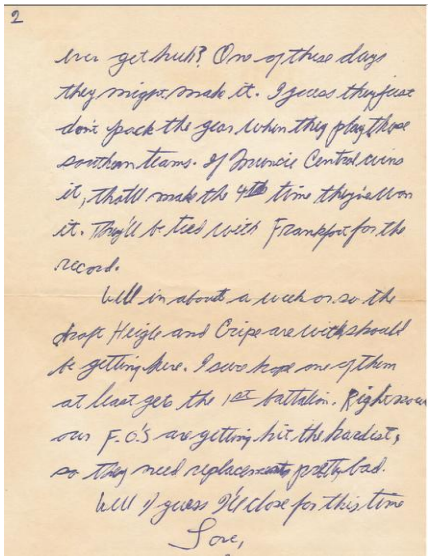

Figure 5: Sample cursive document

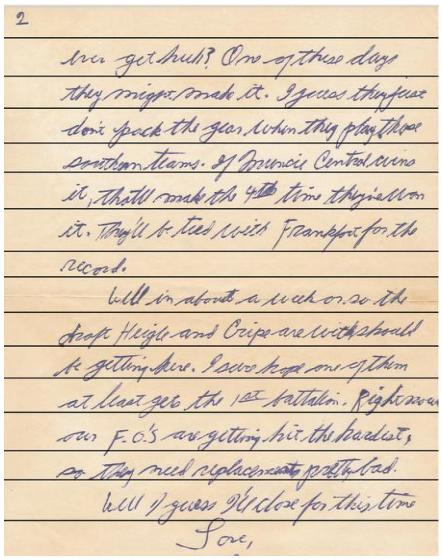

Figure 6: Line breaks detected on sample document

ASCENDERS AND DESCENDERS

Handwritten script, especially cursive, is characterized by ascenders and descenders. An ascender is the top part of a letter that rises above the other letters. This typically occurs with capital letters, some lowercase, or when the writer exaggerates part of a letter beyond the typical boundaries of that letter; examples of lowercase acenders would be ‘l’, ‘t’, ‘b’, ‘d’, ‘f’, ‘h’, and ‘k’. Descenders are similar; these are letters that descend lower than the base of the word to which they belong. The most common descenders in English are lowercase ‘g’, ‘y’, ‘q’, ’p’, and ‘j’. In cursive, this list is extended to include the cursive form of ‘f’. Due to the artistic nature of cursive many other letters are exaggerated and end up being either ascenders or descenders.

I addressed the issue of ascenders and descenders between lines by extending the boundaries of lines after locating initial break points. Without this step, recognized lines excluded several ascenders and descenders, thus resulting in partial letters, which would complicate the task of transcription. For example, there are remarkable similarities between cursive handwritten ‘u’ and ‘y’ as well as ‘n’ and ‘h’. The ascenders and descenders are critical for a human transcriber to distinguish between these letters. This method works very well to give the transcriber a better picture of the full line. An example of the difference is depicted in Figure 7.

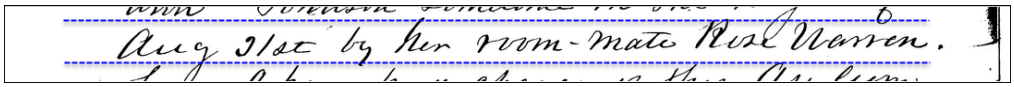

Figure 7: Example of the original line segment boundaries, indicated by the blue lines, versus the extended boundaries.

SHEARING

Most of the cursive writing that I have observed in the CSH dataset is sheared forward, which makes word breaks difficult to detect. In order to detect these word breaks successfully I corrected the forward shear by shearing the line image counterclockwise. Initially, I used the AffineTransform implementation in the skimage transform library. This worked well on smaller images, but when I scaled up to the larger images I encountered issues that prevented the process from completing. After much

troubleshooting and debugging I discovered that there was incompatibility between the binary files that were being produced on the larger images and the shearing algorithm. To rectify this situation I developed and implemented a simple shearing algorithm as follows:

```
def shear(array, strength, shift_axis, increase_axis)
res = np.empty_like(array)
index = np.index_exp[:] * increase_axis
roll = np.roll
for i in range(0, array.shape[increase_axis]):
index_i = index + (i,)
res[index_i] = roll(array[index_i], -i * strength, shift_axis)
return res
```

The script iterates through each row of an image and shifts pixel values by positive or negative integer factors depending on the parameter passed. The script treats each row of pixel values as a list and 'rolls' values to the left to achieve the counterclockwise rotation effect.

WORD SEGMENTATION

With the lines segmented and sheared, a word segmentation algorithm, that I developed and implemented, then detects and segments words from each line and stores them in a database table. Using a method similar to that of line segmentation, the word segmentation algorithm first obtains pixel density for each column within an identified line of text. At this point, the images are represented by a datatype that is specific to the numpy library, one that is compatible with many Python libraries, but is not compatible with Gamera's projection functions. The word segmentation algorithm performs a

vertical sum of pixel values (either 1 for white or 0 for black) down each column of the document to achieve the same effect, a vector of pixel densities per column. Similar to the process for identifying line breaks, the script then performs a local minima search across the width of the line via a sliding window. In the case of word detection I set a requirement in the script that the pixel values within the window have some fluctuation in order for a local minimum to be considered a minimum. I accomplished this by requiring the minimal pixel value in the window to be different than the maximum pixel value. By adding this requirement the occurrence of empty word images, those with no text as in Figure 8, significantly decreased.

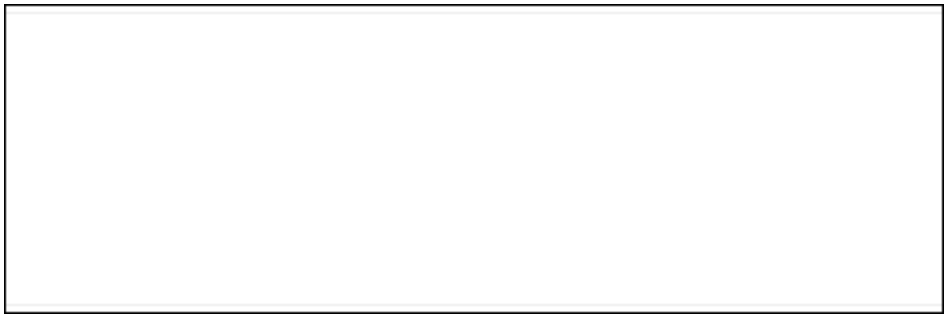

Figure 8: Example of a word image with no text.

Additionally my script requires the local minima to have some area around it that is also empty because there are cases in which thresholding and simple human error during the event of writing can create false word breaks. Figure 9 shows an example of the required width to qualify as a word break.

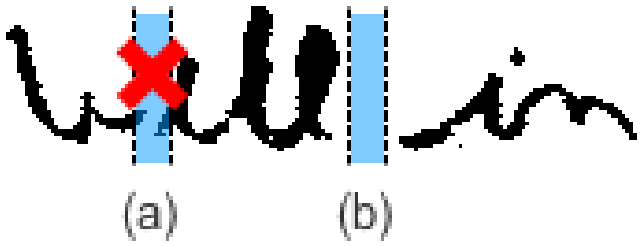

Figure 9: Example of the required pixel threshold needed for a word break to be considered a word break. (a) represents a case that would not be considered a word break. (b) represents a case in which a word break would be generated.

Considering that the documents in the CSH dataset are of high quality with widths around up to and exceeding four thousand pixels, a single pixel break is barely visible so this threshold around word breakpoints is essential in order to avoid false word breaks. The same technique may not be applicable on lower quality images with lower pixel densities. The script also stores the individual word-images in a database table in order to tie them back to the pages, lines, and transcriptions to which they belong.

STEGANOGRAPHY AND TRANSCRIPTION

In order to capture word transcriptions I set up a public transcription portal at https://razor.ischool.utexas.edu/steganoscription. This web interface displays a random word image from the word database. Before being displayed, the script randomly manipulates the image in one of six ways and saves it in a temporary web accessible file. The current manipulations are blue color change plus wave, noise, 60 degree arc, -40 degree arc, red color plus wave, red color change + background speckles. These manipulations are respectively illustrated below in Figure10 – Figure 15.

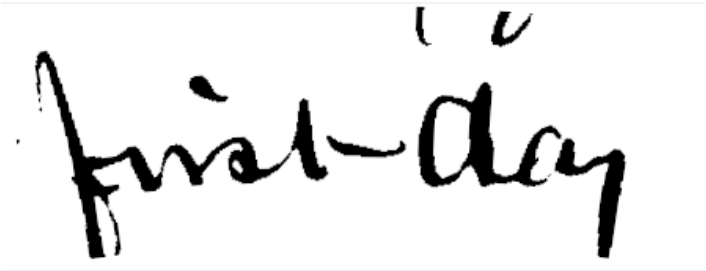

Figure 10: Black text with a small concave arc.

Figure 11: Blue text with an applied geometrics wave.

Figure 12: Black text with a greater concave arc.

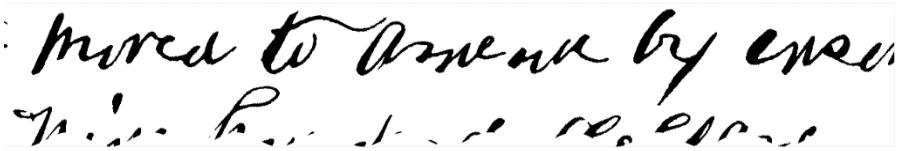

Figure 13: Black text with an applied geometric wave.

Figure 14: Line Red text with an applied geometric wave.

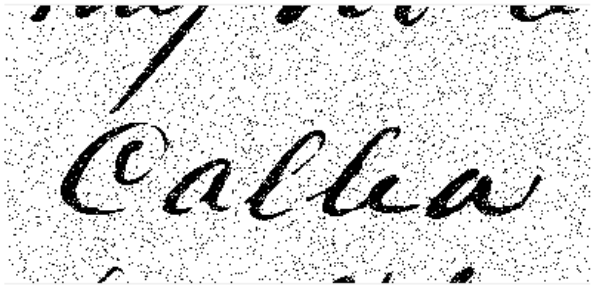

Figure 15: Black text with a noise filter.

The script displays widgets that provide users with three options: insert text into transcription field and submit in order to upload the transcription to the database, click the “No text in image” button if there is indeed no text in the image, or click the “Next image” button if they want to try a different word. The “No text in image” button

provides administrators with data to use for curating the word-image collection. Currently the segmentation scripts do not automatically detect and remove empty images, so this feature allows these images to be flagged by transcribers and later removed by administrators.

Chapter 4 System

System Overview

I built the system specifically for the CSH dataset, but the architecture could be used on any image dataset. The system consists of four database tables and a single dynamic web page. At a high level, the system is very low maintenance. My scripts pre-populate the pages, pg_lines, and line_words tables, which store page images, line images, and word images respectively. The web portal retrieves images from the data set and populates word transcriptions provided by the web site visitors in the fourth database table, word_text. Figure 16 below shows communication within the system.

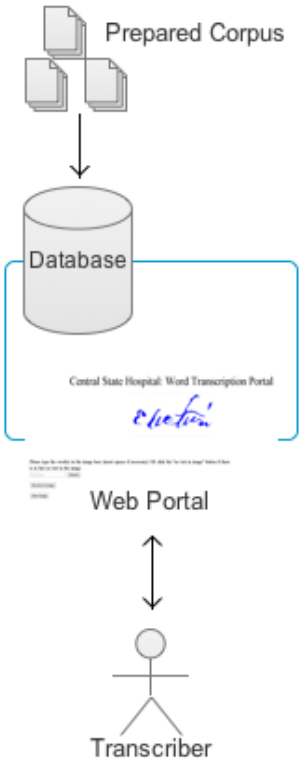

Figure 16: High level system view.

DATABASE SCHEMA

The tables in the database each record information about the documents at various levels. The pages table contains document level information, such as unique page identifier, RGB values, row projection vector for the document, column projection vector for the document, and the actual image file name within the CSH file system. The pg_lines table contains line images for each line within each page. Lines are given a unique ID and also contain the page ID for the page that they belong to as a foreign key. Similarly the line_words contains unique identifiers for each word and the line that they belong to as foreign keys. Both the pg_lines table and line_words tables contain image metadata: x and y position within the original document, height, width, and file name. The word_text table captures the word ID and transcription submitted from the web portal. A new record is created when (1) a user submits a transcription, or (2) a user clicks the "No text in image" button. The text provided by the portal is inserted into the transcription field; if the user clicks the no text button, then a keyword indicating this fact is inserted instead; see Figure 17 for a visual illustration of the schema. The file names saved in the pages, pg_lines, and line_words tables are relative to a base file system path.

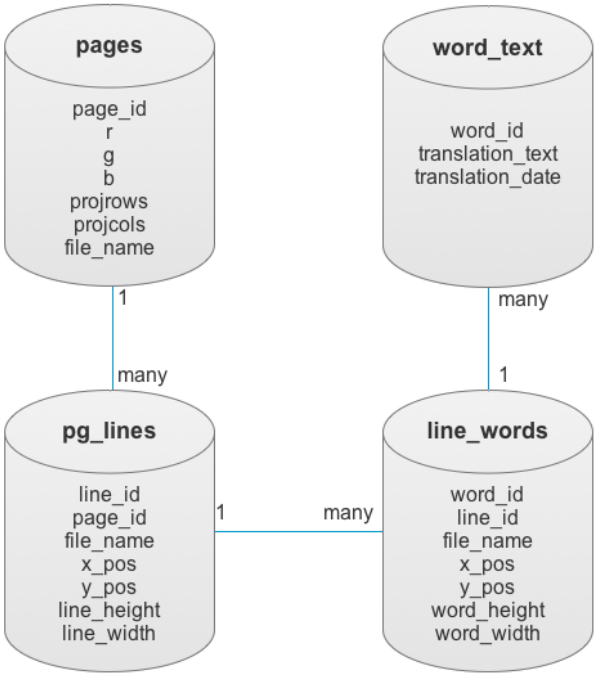

Figure 17: Database schema with table descriptions.

TRANSCRIPTION PORTAL

The web portal, illustrated in Figure 18, is designed for simplicity and with a functional goal. The image is the primary focus, with secondary focus being on the interaction elements below the image. It needs further UI development in order to be aesthetically pleasing.

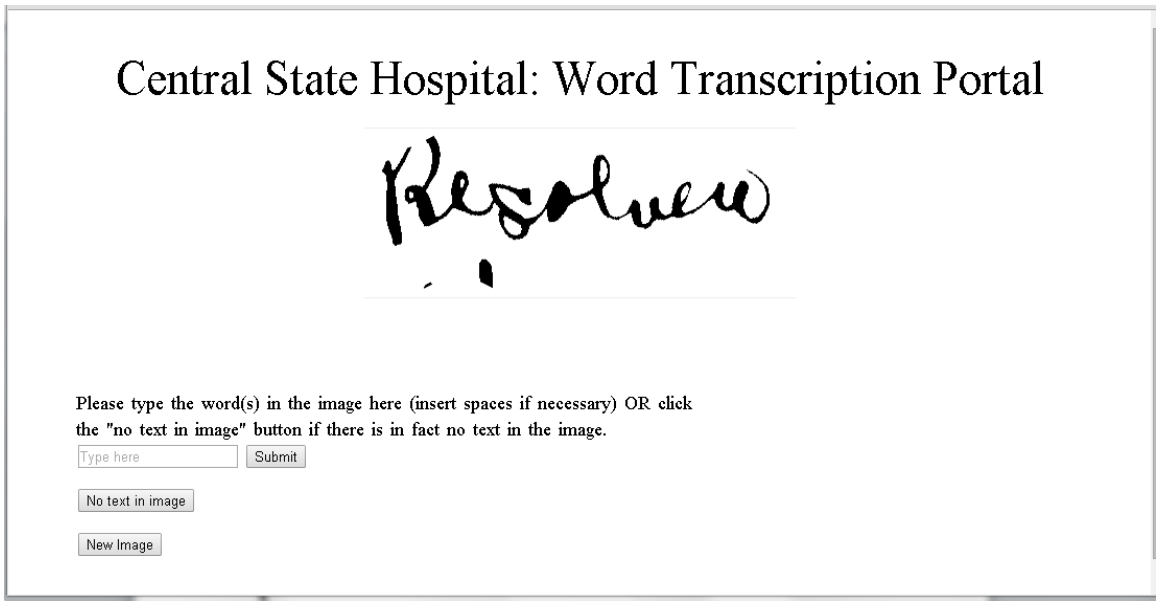

Figure 18: Transcription web portal – the current interface.

The interaction elements are simple with security being at the forefront of each database call. No scripts are run that endanger the file system and any text that is added into the database is stripped of possibly malicious content before query execution. This is done with standard functions designed to prepare user-generated data for input into a database.

Chapter 5 Evaluation

In order to ensure that the algorithms perform satisfactory and accomplish the stated goals of separating words from the documents that include them, I evaluated the developed system components. System components that will be evaluated include: image binary thresholding, line segmentation, line-image shearing, and word segmentation.

IMAGE BINARY THRESHOLDING

I tested multiple thresholding methods before settling on DjVu, specifically Gamera's custom thresholding algorithm and another popular method introduced by Otsu (1979) called Otsu thresholding. In testing, DjVu performed better: producing a clearer image in a similar amount of processing time vs. the Otsu method. Gamera's method is a simple binary threshold that decides if a pixel will be black or white based on a crude method (i.e., pixels greater than X are black and pixels less than X are white), which is not a sufficient solution to the corpus images as it produced subpar binary images that were not suitable for subsequent processes. A key feature of DjVu thresholding is the prevention of "blockiness", which would be important to produce a clearer word when the script is already unclear.

The thresholding step is by far the greatest bottleneck in the system at about 4 minutes of execute time for each full page document. Across the 500,000 CSH images the total execution time for that one step is upwards of 2,000,000 minutes, or 1,400 days of computing time. Obviously that indicates a lack of scalability in the thresholding

algorithm, which was not optimized for the scale of the corpus. With some optimization the processing time could be more manageable, but the likelihood of significant improvement is low considering the size of the images and the processes needed for good thresholding to occur (i.e. pixel clustering and comparison). The more effective option to minimize processing time would be to scale horizontally across multiple computing nodes. There are no dependencies on each document, so the pages could be processed in parallel, but for this proof of concept I proceeded with the smaller problem of processing single documents.

LINE SEGMENTATION

I evaluated the effectiveness of the line segmentation algorithm using the following metrics: error rate for valid line detection, detection legitimacy, and execution time.

Actual line detection refers to the likelihood that when a line is detected, it actually contains a line of text versus a blank line or some partial, illegible line of text. I performed a manual random sampling of about twenty documents, I assessed that about 10% of line images within the Board Minute image set are blanks or incomplete. These occurrences appear for a number of reasons, the primary being a result of how the images were captured originally. The digital master documents include background, the surface on which the documents were resting during the digitization process. While it's important for archival purposes to include the entire document, when this background is binarized it is seen as a black border of uneven size. This causes the line segmenting algorithm to

generate false positives, identifying lines where there are none. If the border were to be removed, the line segmentation algorithm would ignore these empty spaces. It's also difficult to develop rules to ignore the lines because they are non-uniform down the page.

Detecting and segmenting a correct line 90% of the time is an acceptable level of performance for this task. On a test document with no borders, splatters of ink, or slanted lines, the performance was perfect at 0% error for the line detection task, which indicates that the performance depends upon the quality of the digitized images.

Compared to the binary thresholding step, the line segmentation step runs quicker: in less than thirty seconds. The profile of this corpus lends itself to vertically short lines and full pages. On average, there are 53 lines in a Board Minute document. Meaning that in thirty seconds of runtime, the algorithm is detecting each line break and saving a new image file in the file-system for each line. Breaking down the processing time further, the line segmentation algorithm is saving about two line-images per second.

Another tricky aspect of line segmentation in this corpus is the handling of ascenders and descenders in the context of line height. Terminating lines at local minima from one break point to the next inevitably resulted in cases where the resulting lines included several partial letters. In the Board Minutes, acenders and descenders often encroach up to half way into the line above or below. More often than not this occurs with line descenders. I addressed this issue by extending the boundaries of a "line" after locating the midpoint between lines by adding 30% more pixels to the bottom of the segment and 20% more pixels to the top. The approach to the top and bottom differ

because the observed descenders were more exaggerated than the ascenders. To Arrive at 20% and 30%, I tweaked the input until the resulting segments were accurate enough to be readable, but were not so inclusive that a human transcriber would be confused.

This is necessary to keep each word legible for the transcribers. With this modification in place, the algorithm only cuts off small bits of letters in very rare cases. The text that is cut off is not enough to be detrimental to the legibility of the word either.

LINE-IMAGE SHEARING

Compared to one of the best python image manipulation libraries, skimage, my custom shearing function is lacking in features and flexibility. Skimage allows for full 360 degree counterclockwise shearing, while my function can only rotate in 45 degree increments. This meets the needs of the application to most of CSH's board minute registers, but will not be scalable across the entire document set. As writers and the script context changes, there will be cases in which the cursive slant is different or even non-existent. At least one case exists within the board minute documents where the same page contains script with the expected slant as well as script with no slant. In this case my function will also shear the latter case in the same manner as the former case, resulting in script that appears slanted to the left and script that is difficult to segment into words. Currently I am applying one shear angle to all lines, but a very important enhancement moving forward would be adaptive shearing; I'll discuss this process in the next chapter.

While my function lacks granular rotation, it performs more consistently and efficiently than skimage's shear. It performs more consistently in that it runs on small images as well as very large images, while skimage fails at larger resolutions. One advantage of my simpler, no-frills shearing function is that it runs faster as it requires fewer steps to complete processing. In contrast, the shear function in the Skimage library needs to create a special transformation object before executing a warp function to apply the shear.

WORD SEGMENTATION

I evaluate the effectiveness of the word segmentation algorithm in a much the same way that I evaluate the line segmentation: accuracy, execution time, and detection error rate.

Word detection refers to the likelihood that when a word boundary is detected, it contains a word or partial-word of text and is not a false positive, such as a blank image or noise. Completely blank images are very rare or non-existent. In the case of line segmentation, extraneous page borders resulting from the process of digitization were responsible for the blank lines, but with word segmentation there are no borders; so, the algorithm is able to successfully cull blank images directly. It does this by not allowing an image to be saved if the minimum pixel count in the columns is the same as the maximum pixel count in the columns. This occurs in two cases: a completely black image or a completely white image, both of these cases are undesirable.

While removing entirely empty spaces from a line works well, there are numerous cases in which the algorithm mistakes small dark pixel areas for words. Since descenders and ascenders are so prevalent in the writing, it is not uncommon to see only the bottom or top of a letter in a word image. By including such cases, the word segmentation algorithm performs poorly quite often. I have not developed a solution to removing these images yet, other than allowing the system to be curated through the "no text in image button" provided at the transcription portal. Additionally, even when the word segmenter captures text accurately, it doesn't always capture full words. It is difficult to evaluate this positively or negatively because this performance is directly related to the writer's style. There are cases in which two distinct words are run together as if they were one word and there are also cases in which the first letter of a word is given a liberal amount of space between it and the rest of the word. While semantically a single capital letter does not qualify as a word, the algorithm treats it as such, in maintaining fidelity to the nature of the writing. "Fixing" the writing technique algorithmically would compromise the integrity of the writing.

Similar to the line segmenter, the processing time of the word segmenter is very low as a percent of total processing time; it runs in less than ten seconds per line and generates 8 word-images per line on average for the Board Minute registers.

Chapter 6 Discussion and Future Work

In this paper I introduce a technique and a software system for privacy-preserving transcription of handwritten documents through crowdsourcing efforts. Much work has been done over the decade of development in this field, but there has been a lack of focus on enabling the crowdsourcing of handwritten script transcription. In this discussion I want to highlight some cases where my research fits within the community, strengths of the work, weaknesses of the work, other field in which this work can be useful, and how this work can be extended upon with future work.

RESEARCH COMMUNITY CONTEXT

Like my research, Tomai et al. (2002) performs preprocessing steps such as line and word segmentation, and then, unlike my research, maps from an image to a machine-readable word bank based on constraints and feature matching. Their approach requires that a machine-readable word bank exist for whatever data set being used. My work could work well in tandem with their approach; cases in which machine-readable words don't exist for a corpus, those words could then be included in the transcription word image set. Once the word image is transcribed the machine-readable text can then be inserted into the word bank for future use.

While there are strengths to the shearing function I wrote, Dong et al. (2005) approach is presented as more robust and efficient that other algorithms. Unfortunately, there are no Radon transformation-based systems available and the complexity presents a

significant barrier for use, so I was not able to take advantage of this method for my research. Using their method could have improved the effectiveness and versatility of my line shearing by de-shearing each line relative to how forward-sheared the script is originally.

The reCAPTCHA system, presented by Ahn et al. (2008), has enabled large-scale transcription for "old printed material" (Ahn et al. 2008, 1465) and is being used widely across the web, but it does not specifically support handwritten documents. My work could be used to expand the scope of the reCAPTCHA system to include handwritten documents.

STRENGTHS

As reviewed in prior chapters some primary strengths of this work are that it handles cursive handwriting, enables the public transcription of sensitive text, and provides the database infrastructure for scaling up.

While the database schema is not novel, it serves as a contribution directly to the CSH archives. It will enable the CSH team to conduct quantitative analysis on the document set, which was not possible before. Basic query can retrieve telling statistics (i.e. words per line, lines per document, and words per document) and observing these over the life of the documents can reveal implications around the tenure of one writer over another or how the use of words changes over time. For example, did the writer use more or less words per line in 1870 versus 1920? Were they larger or smaller words? By

breaking down the table structures in a way that directly relates pages to lines to words, I have created a simple way to gain insight into these minor changes over time that are accessible via large data sets.

WEAKNESSES

Several weakness of this work include blank images not being removed, a generalized shear function that does not handle special cases or variations in shear intensity, and the lack of support for removing borders around images that disrupt the line segmentation process.

INTERDISCIPLINARY VALUE

Another field that could potentially use this system is the medical industry. Medical technology is rapidly expanding and hospital staffs still maintain several handwritten records. My techniques could reduce cost of filing and storage if these documents could be digitized. They could also be used to digitize existing patient records.

FUTURE WORK

Future work could follow four trajectories, enhance the effectiveness of the algorithms, improve the efficiency of the algorithms, apply crowdsourcing techniques to maximize

the application of human effort, and add features to the Web portal for improved transcriptions.

To enhance the effectiveness of the scripts several improvements can be made. I mentioned adaptive shearing in prior chapters. This feature would de-shear the script on a line by line basis, or even word by word, so that there are no cases in which the script is de-sheared more or less than necessary. This would greatly improve the outcome of the word segmentation script. Another important feature to add would be automated blank image detection and removal. This could be added to the word segmentation process; before saving an image to the file system this feature would check the image for text and not save it if no text is detected. Finally, a feature could be added to the process of line segmentation that detected borders around the document and automatically removed them prior to segmenting lines. This would reduce the number of empty lines to a small number, if not zero, and would thus improve overall effectiveness of all following steps.

As mentioned in chapter 5, processing image documents individually is a time intensive task, and would not be scalable for large data sets. A proven method for scaling computing tasks is to run the same task simultaneously across multiple machines. This type of processing is only possible if there are no dependencies between tasks. Since each document image is processed individually, then there are no dependencies between documents and they can be processed in parallel. Future work should include building the framework for this parallel processing.

To transcribe large data sets, a robust crowdsourcing strategy is necessary. Human transcribers drive the data collection for this system, so scalable crowdsourcing techniques will be a crucial step towards effectively transcribing full data sets. Examples of these techniques include scheduling low cost Human Intelligence Tasks (HITs) on crowdsourcing platforms, incentivizing transcription efforts, or finding ways to spread the system socially or virally.

To empower other archives to use the system and market it for their own collection future work would include an API for connecting to the system and to a hosted database, as well as a javascript portal to replace the static portal so that it could be dispersed as far as possible across the web. This also contributes to methods for scaling the crowdsourcing effort. Making the portal available for anyone to host of a website is key to obtaining transcriptions.

References

Bozinovic, R. M., & Srihari, S. N. (1989). Off-line cursive script word recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *11*(1), 68-83.

Breuel, T. M. (2003, April). High performance document layout analysis. *In Proc. Symp. Document Image Understanding Technology*.

de Zeeuw, F. (2006). Slant Correction using Histograms. *Undergraduate Thesis. http://www. ai. rug. nl/~ axel/teaching/bachelorprojects/zeeuw_slant correction. pdf.*

Dong, J. X., Dominique, P., Krzyyzak, A., & Suen, C. Y. (2005, August). Cursive word skew/slant corrections based on Radon transform. In *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on* (pp. 478-483). IEEE.

Droettboom, M., MacMillan, K., & Fujinaga, I. (2003, April). The Gamera framework for building custom recognition systems. In *Proceedings of the Symposium on Document Image Understanding Technologies* (pp. 275-286). (see also http://gamera.sourceforge.net/)

Ehrich, R. W., & Koehler, K. J. (1975). Experiments in the contextual recognition of cursive script. *IEEE Transactions on Computers*, *24*(2), 182-194.

Fischer, A., Indermühle, E., Bunke, H., Viehhauser, G., & Stolz, M. (2010, June). Ground truth creation for handwriting recognition in historical documents. In

Proceedings of the 9th IAPR International Workshop on Document Analysis Systems (pp. 3-10). ACM.

Guillevic, D., & Suen, C. Y. (1995, August). Cursive script recognition applied to the processing of bank cheques. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on* (Vol. 1, pp. 11-14). IEEE.

Haffner, P., Bottou, L., Howard, P. G., & LeCun, Y. (1999, September). DjVu: Analyzing and compressing scanned documents for Internet distribution. In *Document Analysis and Recognition, 1999. ICDAR'99. Proceedings of the Fifth International Conference on* (pp. 625-628). IEEE.

Hayes, B., Tesar, B., & Zuraw, K. (2003). OTSoft: Optimality Theory Software (Version 2.1) [Software]. Available from http://www.linguistics.ucla.edu/people/hayes/otsoft/

Mermelstein, P., & Eyden, M. (1964, October). A system for automatic recognition of handwritten words. In *Proceedings of the October 27-29, 1964, fall joint computer conference, part I* (pp. 333-342). ACM.

Morkel, T., Eloff, J. H., & Olivier, M. S. (2005, June). An overview of image steganography. In *ISSA* (pp. 1-11).

N. Otsu: *A Threshold Selection Method from Grey-Level Histograms.* IEEE Transactions on Systems, Man, and Cybernetics (9), pp. 62-66 (1979)

Plamondon, R., & Srihari, S. N. (2000). Online and off-line handwriting recognition: a

comprehensive survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, *22*(1), 63-84.

Sayre, K. M. (1973). Machine recognition of handwritten words: A project report. *Pattern Recognition*, *5*(3), 213-228.

Tomai, C. I., Zhang, B., & Govindaraju, V. (2002). Transcript mapping for historic handwritten document images. In *Frontiers in Handwriting Recognition, 2002. Proceedings. Eighth International Workshop on* (pp. 413-418). IEEE.

Vinciarelli, A. (2002). A survey on off-line cursive word recognition. *Pattern recognition*, *35*(7), 1433-1446.

Von Ahn, L., Maurer, B., McMillen, C., Abraham, D., & Blum, M. (2008). recaptcha: Human-based character recognition via web security measures. *Science*, *321*(5895), 1465-1468.

Zimmermann, M., & Bunke, H. (2002). Hidden Markov model length optimization for handwriting recognition systems. In *Frontiers in Handwriting Recognition, 2002. Proceedings. Eighth International Workshop on* (pp. 369-374). IEEE.